基于 Selenium 框架的文献传递机器人的开发与应用——以江苏省工程文献信息中心为例*

申继年[1]， 李静 1，刘新霞[2]

[1]（中国药科大学图书馆，江苏 南京 211198）

[2]（南京财经大学图书馆，江苏 南京 210023）

摘要：[目的/意义]笔者在江苏省工程文献信息中心平台承担文献传递工作，其中机械重复的操作占比很大，严重侵占了工作时间。而且文献需求是 24 小时的，作为值班馆员只能在有限的时间来进行传递，传递时效性难以得到保证。所以，笔者尝试开发一个文献传递机器人，自动化实现完成文献传递工作。[方法/过程] 论文以江苏省工程文献信息中心平台为例，基于 Selenium 框架，设计开发文献传递机器人，实现登录、签到，获取订单、检索、下载、上传，完成文献传递后再继续等待派单的全过程。[结果/结论] 通过一个月的运行分析，文献传递机器人虽然不能完全取代人工传递，但却可以有效解决文献传递过程中的占比很大的机械重复的工作，提高了传递效率，同时，又拓展了服务时间，让传递馆员有更多的精力解决特殊文献等难查找的文献。最后，文献传递机器人对文献传递领域和其他相关场景也有借鉴意义。

关键词：文献传递； 文献传递机器人； Selenium； 江苏省工程文献信息中心

分类号： G252.6

Development and Application of Document Delivery Robot Based on the Selenium Framework

Shen Jinian[1], Li Jing[1], Liu Xinxia[2]

[1]（Library of China Pharmaceutical University，Nanjing 211198, China）

[2]（Library of Nanjing University of Finance and Economics，Nanjing 210023, China）

Abstract: [Purpose/significance]The author is responsible for document delivery work on the Jiangsu Engineering Technology Literature Information Center platform where mechanical repetitive operations account for a large proportion, seriously encroaching on work time. Moreover, the literature requirement is 24 hours, and as a duty librarian, it can only be transmitted within a limited time, making it difficult to ensure the timeliness of transmission. So, the author attempts to develop a document delivery robot to automate the completion of document delivery work.**[Method/process]**The paper takes the Jiangsu Engineering Technology Literature Information Center platform as an example, based on the Selenium framework, designs and develops a document delivery robot, which realizes the entire process of logging in, checking in, obtaining orders, searching, downloading,

---

* 论文系江苏省工程技术文献信息中心基金项目 “江苏省工程技术文献信息资源及共享服务平台共享共建”（项目编号：7422000027/018）研究成果之一

uploading, completing literature transfer, and then continuing to wait for dispatching.**[Result/conclusion]**After a month of operation analysis, it was found that although the document delivery robot cannot completely replace manual delivery, it can effectively solve the high proportion of mechanical repetitive work in the literature delivery process, improve the delivery efficiency, and expand the service time, allowing the delivery librarians to have more energy to solve difficult to find literature such as special literature. Finally, document delivery robots also have reference significance for the field of document delivery and other related scenarios.
Keywords: Document Delivery; Document delivery robot; Selenium; Jiangsu Engineering Technology Literature Information Center

文献传递是指将文献资料按照用户的要求，从图书馆或其他信息机构，传递给用户的一种服务方式。文献传递的方式有很多种，包括邮寄、电子邮件、在线传输等。目前，文献传递已成为图书馆数字化建设的重要组成部分，许多高校图书馆和公共图书馆都提供文献传递服务，并且不断加强数字化文献资源的建设和更新，对其服务不断改进和完善。

1. 研究现状

针对文献传递服务已经有很多相关的研究。这些研究主要集中在文献传递服务的模式、流程、服务质量等方面。2014 年，徐春等在《基于抢答机制下的联合参考咨询服务模式研究》[1]中介绍了“江苏省工程文献信息中心平台”的服务模式和流程，同年，徐春等又在《区域性联合参考咨询平台服务模式实证研究》[2]中从文献传递数量、用户需求、资源分布、平台运行模式、馆员贡献度等方面，详细介绍了“江苏省工程文献信息中心平台”运行情况。

文献传递服务的质量是用户最关心的问题之一，许多研究集中在如何提高文献传递的响应速度和准确率，以及如何增加用户满意度和提高服务质量等方面。2014 年，秦霞在《多平台下文献传递用户评价和用户行为的研究》[3]中对 21 位资深文献传递员进行访谈，总结出影响用户评价的因素，将这些因素与文献传递用户的 4 类行为(选择平台、信息检索、提交申请、获取文献) 一一对应进行分析，最后为文献传递平台管理机构提出建议。2020 年，陆尧等在《北京高校馆际文献传递服务用户兴趣关联研究》[4]中引入关联分析挖掘技术对不同专业用户群体和不同高校用户群体文献传递申请单数据进行分析，实现有效和有针对性的文献推荐服务。2019 年，徐枫等在《文献传递伪需求产生的动因分析及对策研究》[5]中探究文献传递伪需求产生的深层原因，并提出减少文献传递伪需求的对策，以提高图书馆文献传递服务效率，帮助用户高效获取各类文献资源。

随着互联网技术的高速发展，新技术层出不穷，许多新技术应用到文献传递服务中，不仅可以提高服务的效率和质量，还可以拓展服务的范围和深度。2019 年，朱玉强在《微信生态下文献传递机器人研发及其应用》[6]中利用微信公众平台，在不搭建第三方服务器、不动用微信高级接口的前提下，开发文献传递机器人程序，实现用户只需发送一条消息即可获取文献全文。但是，诸如此类与文献传递相关新技术应用的研究还相对较少，所以，笔者尝试把 Selenium 框架应用到文献传递工作当中，通过 Python 调用其接口，实现文献传递服务的自动化。

2. 江苏省工程文献中心平台介绍

随着互联网技术的发展，国内应运而生了多种文献传递系统，优点是基于互联网分布技术，既能统一管理，又便于各馆分级接入，也方便读者提交、便捷地获取文献。文献传

递系统的典型代表是高校用户为主的中国高等教育文献保障系统（简称 CALIS）和中国高校人文社会科学文献中心（简称 CASHL）平台。除这些全国性系统外，各地区图书馆也纷纷成立联盟，开发推出了许多传递系统，典型代表是北京地区高校图书馆文献资源保障体系（BALIS）和江苏省工程技术文献信息中心。

"江苏省工程技术文献信息中心（以下简称平台）"是 2004 年江苏省启动建设的四大科技公共基础服务平台之一，也是江苏区域科技创新的文献信息保障服务平台。平台集成了江苏省科技、文化、教育三大系统的省科技情报研究所、省农科院情报所、省技术监督情报所、南京图书馆、南京大学、东南大学、南京农业大学、中国药科大学、南京医科大学和南京工业大学十家单位现有工程技术文献信息资源，与国家科技图书文献中心和长三角区域文献信息资源的共享合作，以共知共享共建的方式构建文献信息资源保障服务体系，联合向全省开放服务。

文献传递过程涉及三个环节：有文献需求的用户、文献传递平台、值班馆员。基本框架如图 1 所示，用户提交文献需求，通过文献传递平台把文献需求分发给值班馆员，馆员获取到文献信息，通过自己的专业知识和文献资源情况，进行检索，获取资源后，再通过文献传平台提交文献，文献最终通过文献传递平台呈现给用户。



图 1 文献传递系统基本框架

为高效运行，平台施行“自动派单”，即系统自动分配订单，值班馆员排队获取订单，先到先得，限时完成。参与传递工作的值班馆员登录平台后，点击“签到”按钮，按顺序加入服务队列。用户提交的订单形成订单队列，系统按照顺序从订单队列指派订单给服务队列中的馆员，并在网页上弹出派单提醒窗口，值班馆员看到提醒后点击“接单”或者“放弃”。点击“接单”后，该服务馆员自动排在服务队列末尾；点击“放弃”，订单顺位派给下一个排队的馆员，放弃订单的馆员自动排在服务队列末尾。为了保证派单系统稳定、有序、高效运行，平台限定中文订单完成时间为 5 min，外文完成时间为 15 min。如果不能完成订单，可以放弃，被 3 个不同的人放弃的订单则变为难题订单[7]。

当前，文献传递依赖于值班馆员，其处理是否及时决定了文献传递的时效性。文献需求是 24 小时的，而值班馆员只能在有限的时间来进行传递，传递时效性难以得到保证。从事文献传递的馆员一般都是兼职工作，长时间的文献传递也会侵占工作时间，而且文献传递大多数都是机械的重复，完全可以由程序自动化完成。综上所述，即为文献传递机器人的设计初衷。

3. 系统设计与实现

3.1 Selenium 框架的选择

Selenium 是一个自动化测试工具，利用它可以驱动浏览器执行特定的动作，如打开网页，抓取数据等操作。自动化代码会调用 Selenium 框架，创建相应的 http 请求，并发送给 webdriver 浏览器驱动，webdriver 浏览器驱动会对请求进行解析，进而对浏览器进行操控,执行过程如图 2 所示。

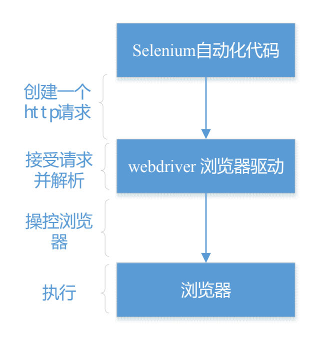


图 2 Selenium 执行过程图

Selenium 与 BeautifulSoup 和 Scrapy 等主流爬虫工具相比，主要区别在于：Selenium 是一个用于 Web 应用程序测试的工具，通过调用相应浏览器的驱动程序，模拟用户进行操作。而爬虫工具主要用于收集数据，爬虫工具发出网络请求时，会通过构造 http 请求头，去指定页面爬取数据，如果网站反爬严格，可以直接识别出爬虫，从而禁止访问。期刊数据库这类网站反爬尤其严格，爬虫工具很容易被限制。Selenium 不属于爬虫，而是仿真测试类框架，直接运行在浏览器中，模拟人工操作，其访问频率和方式不会对目标服务器造成额外负担，也不易被目标网站限制，非常适合需要用户交互的网站，例如登录、提交等各种交互页面。

论文研究的主要问题是自动化完成文献传递整个动作，包括在平台网站领取任务，再去多个检索网站去检索信息、下载并上传。所以，Selenium 框架是最理想的选择。

3.2 系统功能设计

依照平台的文献传递流程，文献传递机器人需实现：平台登录、签到、排队、获取订单、检索、下载、上传，再重新排队整个流程，且全过程无须人工值守，全部由程序自动完成，其详细流程如图 3 如所示。

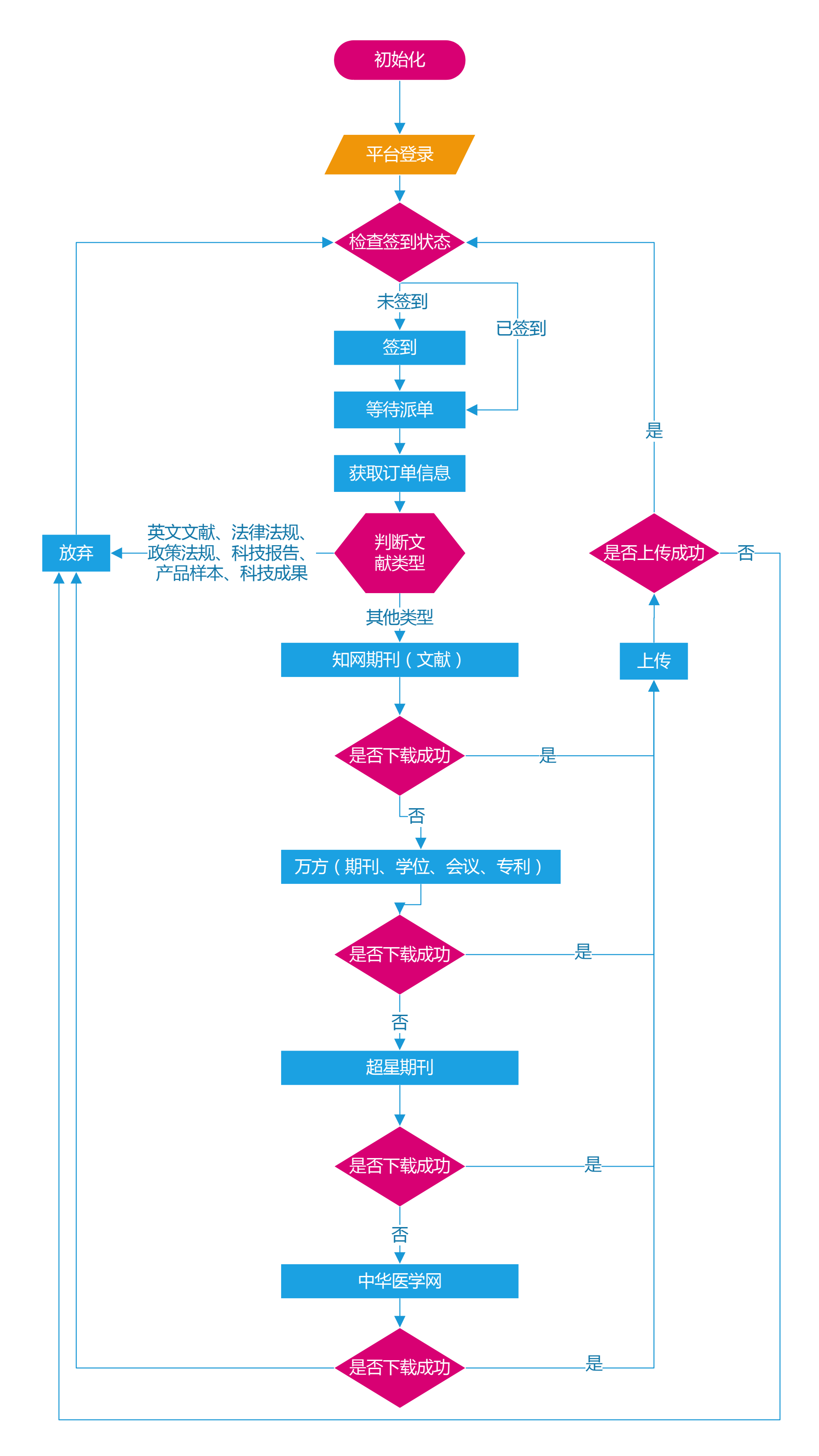


图 3 系统流程图

3.3 功能模块

（1）初始化和平台登陆

初始化主要是程序运行环境的初始化，包括日志、下载路径、运行环境路径、存取订单字典、浏览器驱动器的环境、预先打开的工作窗口等。平台需要值班馆员登录才能领取任务，故在程序初始化完成后，要实现平台登录功能。

（2）签到

值班馆员登录平台后，先签到排队，等待平台分配订单。首次接单或者每次传递结束再接单，需先检测是否签到，如果未签到，则先进行签到，如果已经是签到状态，则直接进行排队，等待派单即可。需要定义一个函数获取签到状态，签到状态如果获取成功，则返回签到状态；如果获取失败，则返回 False。关键代码如下：

```
def check_sign_status():
   try:
      locator = (By.XPATH, '//*[@id="signBut"]/a')
      WebDriverWait(driver, 10, 0.3).until(EC.presence_of_element_located(locator))
      sign_text = driver.find_element(By.XPATH, '//*[@id="signBut"]/a').text
   except Exception as e:
      return False
   else:
      return sign_text
```

（3）等待派单

如果已签到，则交由接单函数等待派单，利用 retry 函数，每隔 5S 尝试一次，直到获取订单。关键代码如下：

```
@retry(wait_fixed=5000)
def order_prompt():
   try:
      driver.find_element(By.XPATH, "//*[@id='orderPrompt']/p[2]/a[1]").click()
   except Exception as e:
      …
```

如果未签到，则交由签到函数进行签到，再等待派单。

（4）获取订单

在获取订单前，先定义一个字典，用于存放订单编号、订单标题、文献类型、作者、来源等订单信息。字典表头如下：

header = ['订单编号', '订单标题', '作者', '文献类型', '来源', '状态', '获取订单时间', '处理完成时间']

再定义一个函数，用于获取订单信息。关键代码如下：

```
def delivery_info():
   delivery_info_dict = dict()
   try:
      locator = (By.XPATH, '//*[@id="deliveryinfo"]/div[2]/div[1]/h1/a')
      WebDriverWait(driver, 5, 0.5).until(EC.presence_of_element_located(locator))
      delivery_info_dict['订单编号'] = driver.find_element(By.XPATH,
                                          '//*[@id="deliveryinfo"]/div[2]/div[1]/p/span').text
      delivery_info_dict['订单标题'] = driver.find_element(By.XPATH,
```

```
                                    '//*[@id="deliveryinfo"]/div[2]/div[1]/h1/a').text
        delivery_info_dict['文献类型'] = \
            driver.find_element(By.XPATH, '//*[@id="deliveryinfo"]/div[2]/div[2]/p[2]').text.split("：")[-1]
        delivery_info_dict['作者'] = driver.find_element(By.XPATH,
                                    '//*[@id="deliveryinfo"]/div[2]/div[2]/p[4]/span').text
        delivery_info_dict['来源'] = driver.find_element(By.XPATH, '//*[@id="deliveryinfo"]/div[2]/div[2]/p[7]/a').text
    except Exception as e:
        return False
    else:
        return delivery_info_dict
```

（5）文献类型的判断

如果获取订单为全英文，或者文献类型为科技成果、政策法规、产品样本、科技报告和法律法规，则直接放弃，结束本次循环，其余订单向下进入检索流程。

判断订单是否为英文文献的方法是检查标题是否是全英文标题，如果是全英文标题，则被视为英文文献，直接放弃。关键代码如下：

```
def is_contains_chinese(strs):
    for _char in strs:
        if '\u4e00' <= _char <= '\u9fa5':
            return True
    return False
```

为了加快订单的处理效率，文献类型为科技成果、政策法规、产品样本、科技报告和法律法规的订单，直接放弃，避免进入检索流程，消耗检索时间。关键代码如下：

```
        if (is_contains_chinese(title) is False) \
                or (delivery['文献类型'] == '科技成果') \
                or (delivery['文献类型'] == '政策法规') \
                or (delivery['文献类型'] == '产品样本') \
                or (delivery['文献类型'] == '科技报告') \
                or (delivery['文献类型'] == '法律法规'):
            delivery['状态'] = '放弃'
            abandon()
            dict_to_csv(delivery)
            continue
```

（6）检索和下载

文献类型过滤结束，订单将依次到四个数据库进行检索。如果检索到结果并下载成功，将直接跳转到上传流程；如果未检索到结果或检索到结果但没权限下载，则放弃订单，结束本次循环。

以 CNKI 为例，把 CNKI 高级检索页面作为检索入口，首先选择下拉菜单并选择“篇名”，然后输入“订单标题”，默认“精确”匹配，最后点击“检索”按钮。检查是否检索的匹配文献，如果检索到文献则下载，否则返回 False。关键代码如下：

```
def get_cnki(delivery_title):
    try:
        locator = (By.XPATH, '/html/body/div[2]/div[3]/div[3]/div[2]/div[1]/div[9]')
        WebDriverWait(driver, 10, 0.3).until(EC.presence_of_element_located(locator))
        search = Select(
            driver.find_element(By.XPATH,
'/html/body/div[2]/div[3]/div[3]/div[2]/div[1]/div[2]/span[2]/div/select'))
        search.select_by_value("TI")
        driver.find_element(By.ID, 'txt_1_value1').send_keys(delivery_title)
        driver.find_element(By.XPATH, '/html/body/div[2]/div[3]/div[3]/div[2]/div[1]/div[9]').click()
    except Exception as e:
        return False
    else:
        try:
            locator = (By.XPATH, '//*[@id="gridTable"]/div/div[2]/table/tbody/tr/td[9]/a[1]')
            WebDriverWait(driver, 10, 0.3).until(EC.presence_of_element_located(locator))
            driver.find_element(By.XPATH,
                         '//*[@id="gridTable"]/div/div[2]/table/tbody/tr/td[9]/a[1]').click()
        except Exception as e:
            return False
        else:
            sleep(5)
```

（7）上传

订单文献下载成功后，程序将自动切换到平台的订单上传页面，点击“立即上传”按钮，完成订单上传。由于平台的上传对话框是调用的 Windows 系统的窗口，所以，实现上传需要调用 Python 的 Win32 库，Win32 是一种在 Windows 操作系统中开发 Python 应用程序的工具包，包含了 Win32API 的模块、拓展类型以及一些辅助工具，开发者可以通过这个包访问到操作系统底层的 API 接口或者编写 Windows 图像界面的应用程序等等功能。关键代码如下：

```
def upload(file_title):
    try:
        locator = (By.XPATH, '//*[@id="deliveryinfo"]/div[2]/div[4]/a[1]')
        WebDriverWait(driver, 10, 0.5).until(EC.presence_of_element_located(locator))
        driver.find_element(By.XPATH, '//*[@id="deliveryinfo"]/div[2]/div[4]/a[1]').click()
    except Exception as e:
        …
    else:
        sleep(2)
        try:
            locator = (By.XPATH, '/html/body/div[9]/div[1]/div[2]/div/div[1]/div/form')
            WebDriverWait(driver, 10, 0.5).until(EC.presence_of_element_located(locator))
```

```
            driver.find_element(By.XPATH,
'/html/body/div[9]/div[1]/div[2]/div/div[1]/div/form').click()
        except Exception as e:
            …
        else:
            sleep(3)
            try:
                window = win32gui.FindWindow("#32770", "打开")
                ComboBoxEx32 = win32gui.FindWindowEx(window, 0, "ComboBoxEx32", None)
                ComboBox = win32gui.FindWindowEx(ComboBoxEx32, 0, "ComboBox", None)
                Edit = win32gui.FindWindowEx(ComboBox, 0, 'Edit', None)
                Button = win32gui.FindWindowEx(window, 0, 'Button', "打开(&O)")
                win32gui.SendMessage(Edit, win32con.WM_SETTEXT, None, file_title)
                sleep(2)
                win32gui.SendMessage(window, win32con.WM_COMMAND, 1, Button)
            except Exception as e:
                …
            else:
                sleep(30)
                return True
```

3.4 源数据库的选择

平台订单的文献类型主要有十种，分别为期刊文章、学位论文、专利、会议论文、科技成果、政策法规、产品样本、科技报告、法律法规、专著。中文文献比较集中，主要来源于中国知网（以下简称 CNKI）、万方数据（以下简称万方）两大中文数据库；外文文献则较分散，大部分来源于：SpringerLink、ScienceDirect、Wiley 等，另外，还有少量文献来自 EBSCO、ACS、ProQuest、IEEE、OXFORDJOURNALS 等外文全文数据库。

由于英文文献来源多且分散，笔者单位所购买的外文数据库种类有限，另外，国内访问外文数据库延迟较大，且易出错，暂时放弃外文数据库的检索。

笔者单位购买的中文数据库子库包括期刊文章、学位论文、专利、会议论文这四大类，科技成果、政策法规、产品样本、科技报告和法律法规没有购买下载权限。所以，中文文献的处理包括期刊文章、学位论文、专利、会议论文四类，其余为科技成果、政策法规、产品样本、科技报告和法律法规等文献类型的订单直接放弃检索。

综上所述，结合笔者单位购买数据库情况，检索的源数据库选定为四个，分别是 CNKI、万方、超星期刊(以下简称超星)和中华医学期刊(以下简称中华医学)。其中，CNKI 和万方文献的重叠率很高，包含了期刊论文、学位论文、会议论文和专利；中华医学主要包含了医学相关文献；超星主要作为前三种期刊的补充。故检索的先后顺序为：CNKI、万方、超星和中华医学。

4. 运行与分析

4.1 运行总体情况

系统自 2022 年 11 月上线测试，已平稳运行一年，论文截取 2023 年 11 月的订单数据进行分析展示，其总体概况如表 1 所示。

表 1 11 月订单完成概况

订单状态	数量
获取订单	8261
完成的订单	3392
未完成的订单	4869
未完成订单中的英文订单	1735

11 月份，笔者账号在平台上共获取订单 8261 篇，其中完成了 3392 篇，完成占比为 41.06%。余下 4869 篇为未完成订单，未完成订单主要有以下三种情况。

第一，英文文献。由于国外数据库的网速不稳定，经常导致检索下载失败，故获取订单时，先判断是否未英文文献，如果为英文文献，会第一时间放弃，继续下一个订单的排队。主动放弃的英文订单为 1735 篇，英文订单占总订单比例为 21%。

第二，检索不到的文献。按订单标题先后在 CNKI、万方、超星和中华医学四个数据库中检索，如果四个数据库种都没有检索到，则放弃本次传递，继续下一个订单的排队。

第三，可以检索到，但无权下载的文献。订单文献可以检索到，但由于笔者所在单位未购买相应的版权，无法下载，则放弃本次传递，继续下一个订单的排队。

4.2 来源数据库分布

11 月份，8261 篇订单来源数据库分布如图 4 所示，其中 23%的订单在 CNKI 中完成，11%的订单在万方中完成 ，1%的订单在超星中完成，6%的订单在中华医学中完成，最后 59%的订单在四个数据库中都无法获取到，选择放弃。

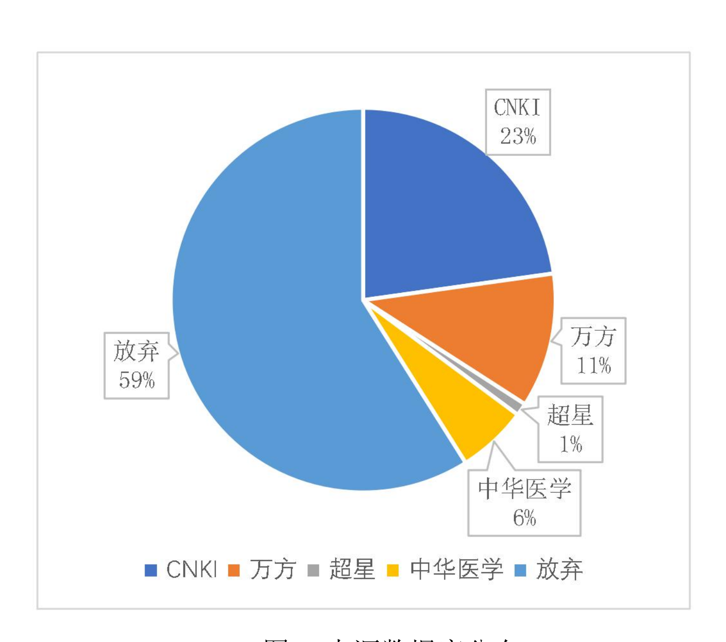


图 4 来源数据库分布

4.3 每日获取订单与完成情况

11 月份，每日获取订单数量和完成率如图 5 所示，平均每日获取订单 275 篇，平均完成订单 113 篇，完成率稳定在 41%左右。

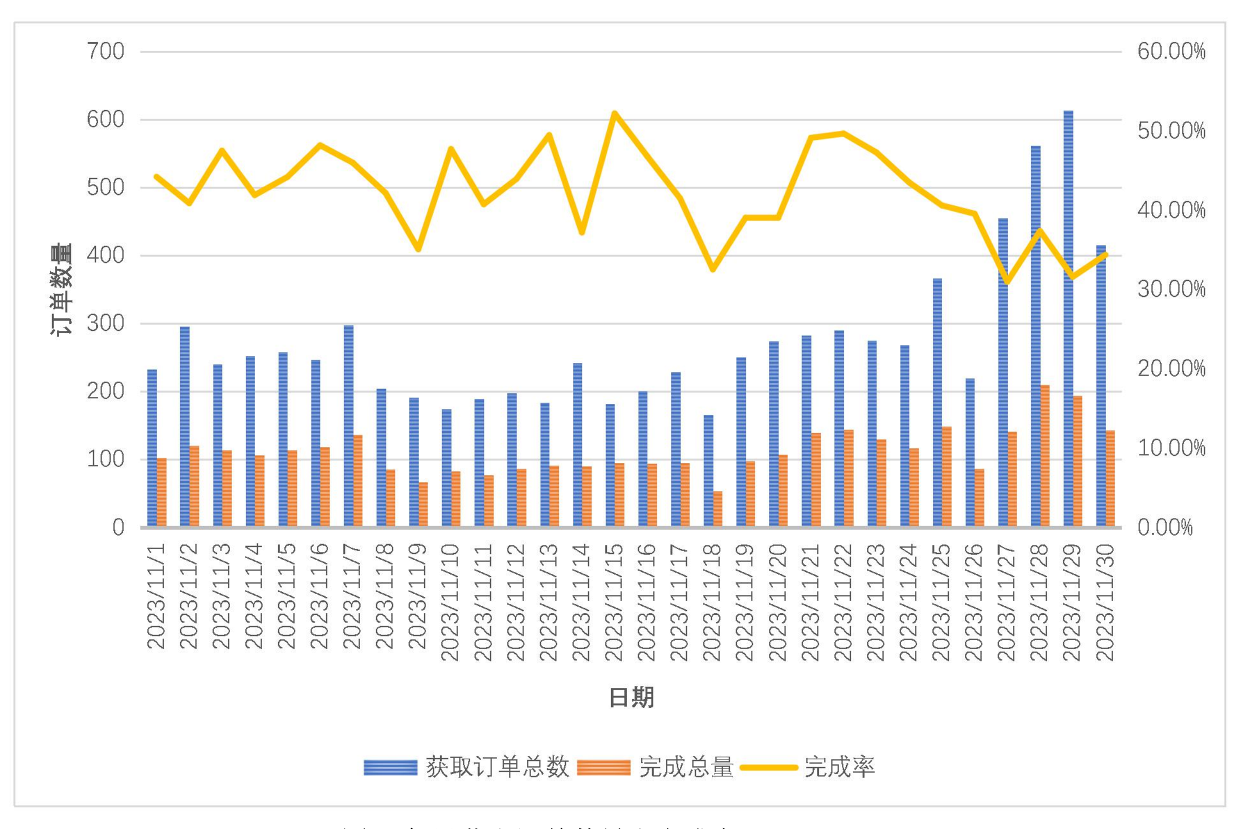


图 5 每日获取订单数量和完成率

4.4 时间段统计与分析

11 月份，订单时间段统计如图 6 所示，订单需求集中分布在 8-22 点之间，其中呈现两次高峰，分别在 10-11 点、15-17 点，订单需求的时间段分布符合科研人员的作息时间。另外，从图 6 也可以看出，订单需求是 24 小时的，尤其是非工作时间，文献传递机器人都能做到第一时间相应，保证了传递的时效性。

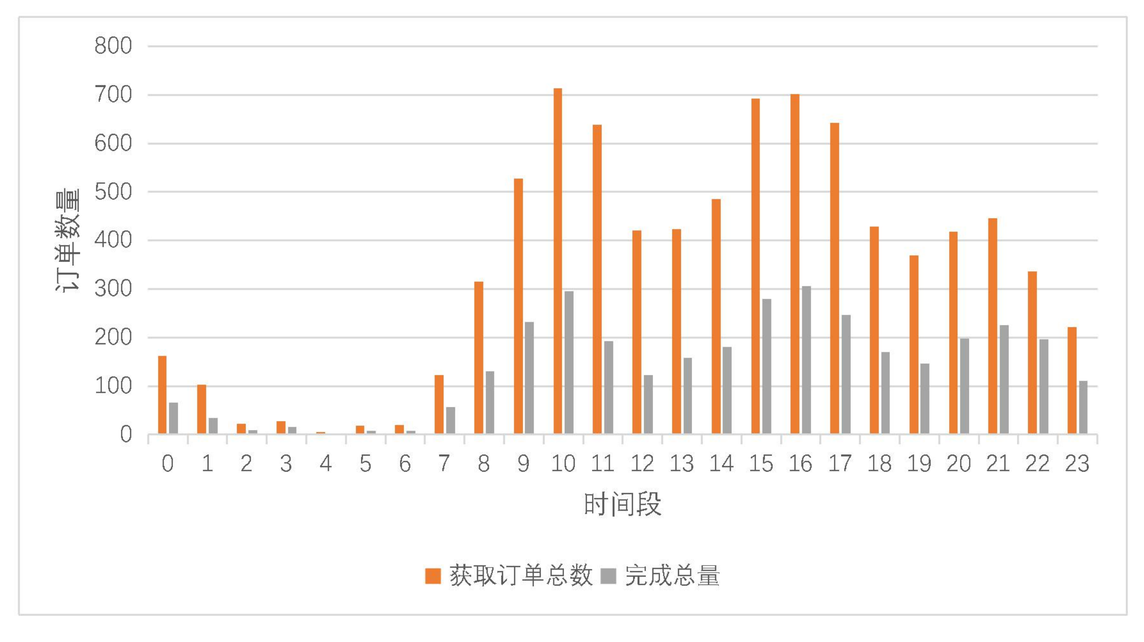


图 6 按时段统计的订单获取和完成量分布图

4.5 订单处理时长与分析

11 月份，订单处理时长统计如图 7 所示，其中处理时长为 0 的订单有 1858 篇，包括

1735 篇英文订单，余下 123 篇为异常订单，直接放弃，未进出处理流程，故耗时为 0。正常的 6403 篇订单的处理时长集中分布在 41-81 秒之间，其中出现 5 次高峰，分别对应图 3 流程图中四个数据库的五个检索下载阶段，先后分别为 CNK、万方期刊、万方专利、超星和中华医学。平台要求订单处理时长不超过 5 分钟，最长订单为 2 分 30 秒，不但满足平台对订单处理时长的要求，在加速订单处理的同时，还解放了值班馆员的时间。

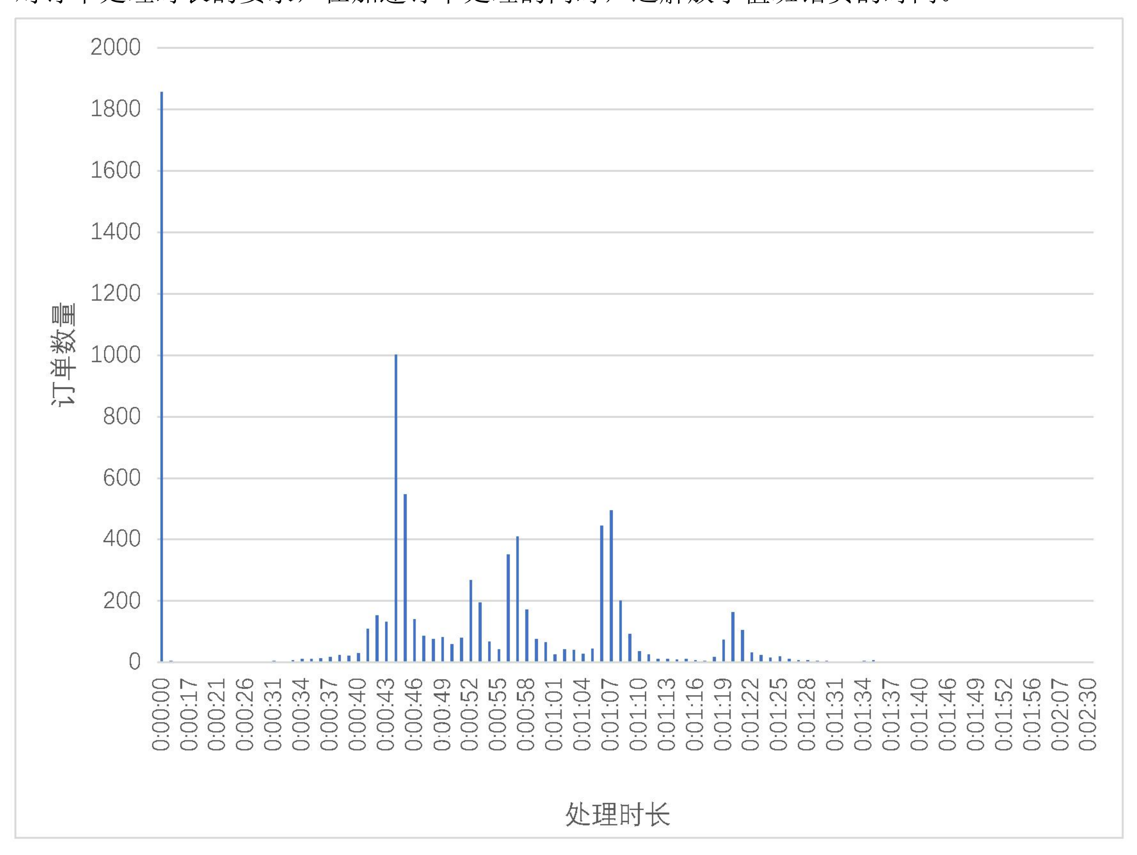


图 7 按处理时长订单分布图

5. 结语

目前，文献传递机器人还不能完全取代人工传递，原因如下：

（1） 用户提交的订单信息不标准，标题和数据库检索到的信息不能完全匹配。

（2） 文献复杂，需要人工核对，比如，简单的标题可能会检索出多个结果。

这些情况导致简单的标题匹配并不能完全满足订单的要求，需要加入人工的判断和纠错，才能保证每一笔订单的顺利完成。

尽管如此，文献传递机器人仍可在文献传递中发挥重要作用，不仅可以提高文献传递效率，节省人力投入，还可以扩展服务时间。建议工程文献中心改进文献派单策略，普通订单先交由各个高校的文献传递机器人自动完成，所有单位都检索失败后，再进入难题库，交由各个单位的传递馆员人工查找。这样既节省了人力，也提高了传递效率，让传递馆员有更多的精力放在英文文献等难查的文献上，而不是一味的追求传递的文献量。以此为例，文献传递机器人也可以推广到更多的文献传递领域和相关场景。

参考文献：

[1]徐春,殷铭.基于抢答机制下的联合参考咨询服务模式研究——以"江苏省工程文献信息中心平台"为例[J].图书馆学研究，2014(3)：80-83.

[2] 徐春,殷铭.区域性联合参考咨询平台服务模式实证研究--以"江苏省工程文献信息中心平台"为例[J].

图书馆理论与实践, 2015(2)：102-105.
[3]秦霞.多平台下文献传递用户评价和用户行为的研究[J].图书情报工作, 2014, 58(18)：41-49.
[4]陆尧,刘春鸿,杨代庆.北京高校馆际文献传递服务用户兴趣关联研究[J].情报理论与实践, 2020, 43(4)：101-107.
[5]徐枫，周秀霞．文献传递伪需求产生的动因分析及对策研究——以东北师范大学图书馆为例［Ｊ］．图
书馆工作与研究，2019(8)：66-70.
[6]朱玉强. 微信生态下文献传递机器人研发及其应用[J].图书馆论坛，2019(9)：135-139.
[7]范闯,殷铭,王飞,等.派单系统在江苏省工程技术文献信息中心平台中的应用[J].江苏科技信息，2021(9)：56-58.

(通讯作者: 申继年　E-mail:shen@cpu.edu.cn)

作者贡献声明：

申继年：提出研究思路，设计开发方案，撰写论文；
李静：测试并提出改进意见；
刘新霞：数据处理并制作图表。